KERJA PRAKTIK

Visualisasi Data dan Evaluasi Kinerja Algoritma Plugin Reasonal

Reasonal DE GmbH

DIREKTORAT PENDIDIKAN INSTITUT TEKNOLOGI SEPULUH NOPEMBER SURABAYA

Periode: 12 Oktober 2020 - 3 Desember 2020

Oleh:

Donny Fitrado 05111740000171

Pembimbing Jurusan
Ary Mazharuddin S., S.Kom., M.Comp.Sc.

Pembimbing Lapangan
Dr. Natalie Tillack

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2020

KERJA PRAKTIK

Visualisasi Data dan Evaluasi Kinerja Algoritma Plugin Reasonal

Reasonal DE GmbH

DIREKTORAT PENDIDIKAN INSTITUT TEKNOLOGI SEPULUH NOPEMBER SURABAYA

Periode: 12 Oktober 2020 - 3 Desember 2020

Oleh:

Donny Fitrado 05111740000171

Pembimbing Jurusan
Ary Mazharuddin S., S.Kom., M.Comp.Sc.

Pembimbing Lapangan
Dr. Natalie Tillack

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2020

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

KERJA PRAKTIK

Visualisasi Data dan Evaluasi Kinerja
Algoritma Plugin Reasonal

Oleh:

Donny Fitrado 05111740000171

Mengetahui,

Pembimbing Kerja Praktik

Dr. Natalie Tillack

Menyetujui,

Dosen Pembimbing

Kerja Praktik

Ary M. Shiddiqi, S.Kom., M.Comp.Sc., Ph.D

NIP. 19810620 200501 1 003

SURABAYA

Desember, 2020

[Halaman ini sengaja dikosongkan]

Visualisasi Data dan Evaluasi Kinerja Algoritma Plugin Reasonal

Nama Mahasiswa	**: Donny Fitrado**
NRP	**: 05111740000171**
Departemen	**: Informatika FTEIC-ITS**
Pembimbing Jurusan	**: Ary Mazharuddin S., S.Kom., M.Comp.Sc.**
Pembimbing Lapangan	**: Dr. Natalie Tillack**

ABSTRAK

Reasonal DE GmbH adalah sebuah perusahaan startup yang bergerak di bidang layanan dan teknologi informasi. Perusahaan ini didirikan berdasarkan topik penelitian dari kedua pendirinya, yang dulu sedang menempuh jenjang studi doktor. Reasonal DE GmbH menyediakan layanan plugin pada browser seperti google chrome, yang dapat digunakan untuk menyimpan *entries* atau *logs* dari suatu artikel. Entry berisi informasi tentang penerbit, indikator konsistensi dalam menautkan ke konten penerbit di website, dan informasi relevan lainnya tentang konten itu sendiri. Plugin ini dapat dipasang dan dipakai secara gratis oleh siapapun.

Meskipun plugin telah berfungsi dengan baik, Reasonal tetap berusaha untuk memperbaiki dan mengembangkan plugin tersebut. Untuk mengetahui perkembangan dari plugin ini, mereka membutuhkan statistik mengenai pemakaian, serta performa dari plugin Reasonal. Statistik pemakaian meliputi sumber penggunaan alamat website *entry* yang dibuat Reasonal, berapa banyak *entries* yang dibuat oleh seorang pengguna, hingga *response time* dari plugin Reasonal ketika digunakan.

Oleh karena itu, dibutuhkan evaluasi kinerja algoritma plugin Reasonal, serta membuat visualisasi data yang berisi performa maupun detail pemakaian pengguna plugin. Visualisasi data akan menggunakan metode bar graph, lalu mengevaluasi kinerja berdasarkan graph yang

dihasilkan. Saya membuat graph dengan menggunakan bahasa pemrograman python yang dikerjakan di website google colab, sehingga dapat mempermudah pengecekan berkala dari pembimbing lapangan. Data yang digunakan telah disediakan oleh perusahaan, dan tidak diperkenankan untuk disebarluaskan ke publik.

Kata Kunci: Python, Plugin, Entry, Workspace, Collaborators

KATA PENGANTAR

Puji syukur kami haturkan kepada Allah SWT karena berkat rahmat-Nya kami dapat melaksanakan salah satu kewajiban kami sebagai mahasiswa Departemen Informatika, yakni Kerja Praktik (KP).

Kami menyadari masih ada kekurangan baik dalam pelaksanaan kerja praktik maupun penyusunan buku laporan ini. Namun, kami berharap buku laporan ini dapat menambah wawasan pembaca dan dapat menjadi sumber referensi. Kami mengharapkan kritik dan saran yang membangun untuk kesempurnaan buku laporan kerja praktik ini.

Melalui buku ini, kami juga ingin menyampaikan rasa terima kasih kepada orang-orang yang telah membantu, baik langsung maupun tidak langsung, dalam pelaksanaan kerja praktik hingga penyusunan laporan. Orang-orang tersebut antara lain adalah:

1. Kedua orang tua penulis.
2. Bapak Ary Mazharuddin Shiddiqi S.Kom., M.Comp., Ph.D, selaku dosen pembimbing dan koordinator Kerja Praktik.
3. Ibu Dr. Natalie Tillack, selaku pembimbing lapangan Kerja Praktik.

Surabaya, 29 Desember 2020

Donny Fitrado

DAFTAR ISI

LEMBAR PENGESAHAN 5

ABSTAK 8

KATA PENGANTAR 10

DAFTAR ISI 11

DAFTAR GAMBAR 14

DAFTAR TABEL 15

DAFTAR KODE 15

BAB I
PENDAHULUAN

1.1. Latar Belakang 17
1.2. Tujuan 18
1.3. Manfaat 18
1.4. Rumusan Masalah 18
1.5. Lokasi dan Waktu Kerja Praktik 19
1.6. Metodologi Kerja Praktik 19
1.7. Sistematika Laporan 21

BAB II
PROFIL PERUSAHAAN

2.1. Profil Reasonal DE GmbH 24
2.2. Logo Perusahaan 25

2.3. Visi Misi Perusahaan 26
2.4. Struktur Organisasi 26

BAB III
TINJAUAN PUSTAKA

3.1. Python 28
3.2. Numpy 29
3.3. Pandas 30
3.4. DataFrame 31
3.5. Matplotlib 35
3.6. Seaborn 38
3.7. Google Colab 40

BAB IV
IMPLEMENTASI SISTEM

4.1. Menyambungkan ke Google Drive 44
4.2. Import Library 44
4.3. Memuat Data ke Google Drive 45
4.4. Memunculkan Dataframe 45
4.5. Unique Users & Referrer 47
4.6. Unique Users & org_id 48
4.7. Unique Users & Browsers 49
4.8. Unique Users & OS 51
4.9. Jumlah Entri di Setiap Workspace 52
4.10. Entri dari Setiap Website 53

4.11. Edits per Entry 54

BAB V
PENGUJIAN DAN EVALUASI

5.1. Tujuan Pengujian 57
5.2. Kriteria Pengujian 57
5.3. Skenario Pengujian 57
5.4. Evaluasi Pengujian 63

BAB VI
KESIMPULAN DAN SARAN

6.1. Kesimpulan 64
6.2. Saran 64

DAFTAR PUSTAKA 65

LAMPIRAN 66

BIODATA PENULIS I 71

DAFTAR GAMBAR

Gambar 2.2.	Logo Reasonal DE GmbH
Gambar 2.4.	Struktur Organisasi Reasonal DE GmbH
Gambar 3.2.	Operasi Penjumlahan 2 List
Gambar 3.4.1	Pembentukan Objek DataFrame
Gambar 3.4.2.	Penggunaan Fungsi Shape
Gambar 3.4.3.	Fungsi head() dan tail() untuk Menampilkan Data
Gambar 3.4.4.	Fungsi Values dan Menghasilkan Array 2D
Gambar 3.5.1.	Visualisasi Matplotlib oleh Nicolas P. Rougier
Gambar 3.5.2.	Pilihan Informasi Visual dalam Matplotlib
Gambar 3.5.3.	Mengimport matplotlib ke Notebook
Gambar 3.6.1.	Pembuatan Barplot dengan Menggunakan Matplotlib
Gambar 3.6.2.	Hasil Barplot Matplotlib
Gambar 3.6.3.	Penggunaan Barplot Seaborn
Gambar 3.6.4.	Hasil Barplot Menggunakan Seaborn
Gambar 4.5.	Grafik Unique Users di Setiap Referrer
Gambar 4.6.	Grafik Unique Users per org_id
Gambar 4.7.	Grafik Unique Users per Browser

Gambar 4.8. Grafik Unique Users per OS
Gambar 4.9. Grafik Jumlah Entri di Setiap Workspace
Gambar 4.10.1. Jumlah Entri dari Setiap Website
Gambar 4.10.2 Grafik Jumlah Entri per Website
Gambar 4.11. Grafik Jumlah Edit per Entri
Gambar 5.3.1 Pengujian Grafik Users dan Referrer
Gambar 5.3.2. Pengujian Grafik Users per org_id
Gambar 5.3.3. Pengujian Grafik Unique Users per Browser
Gambar 5.3.4. Pengujian Grafik Unique Users per OS
Gambar 5.3.5. Pengujian Grafik Entri per Workspace
Gambar 5.3.6. Pengujian Grafik Entri Berdasarkan Asal
Gambar 5.3.7. Pengujian Grafik Edit per Entri

DAFTAR TABEL

Tabel 4.4 Tabel Dataframe
Tabel 4.11. Tabel Jumlah Edit di Setiap Link
Tabel 5.4. Tabel Evaluasi Pengujian Grafik

DAFTAR PSEUDOCODE

Pseudocode 4.1 Menyambungkan ke Google Drive

Pseudocode 4.2 Mengimport Semua Library yang Dibutuhkan
Pseudocode 4.3 Memuat Data dari Google Drive
Pseudocode 4.4 Memunculkan Tabel Dataframe
Pseudocode 4.5 Membuat Grafik Unique Users di Setiap Referrer
Pseudocode 4.6 Membuat Grafik Unique Users per org_id
Pseudocode 4.7 Membuat Grafik Unique Users per Browsers
Pseudocode 4.8 Membuat Grafik Unique Users per OS
Pseudocode 4.9 Membuat Grafik Jumlah Entri per Workspace
Pseudocode 4.10 Membuat Grafik Jumlah Entri per Website
Pseudocode 4.11 Membuat Grafik Jumlah Edit per Entri

BAB I

PENDAHULUAN

1.1. Latar Belakang

Reasonal DE GmbH adalah sebuah perusahaan startup yang bergerak di bidang layanan dan teknologi informasi. Perusahaan ini didirikan berdasarkan topik penelitian dari kedua pendirinya, yang dulu sedang menempuh jenjang studi doktor. Reasonal DE GmbH menyediakan layanan *plugin* pada *browser* seperti google chrome, yang dapat digunakan untuk menyimpan *entries* atau *logs* dari suatu artikel. *Entry* berisi informasi tentang penerbit, indikator konsistensi dalam menautkan ke konten penerbit di website, dan informasi relevan lainnya tentang konten itu sendiri. *Plugin* ini dapat dipasang dan dipakai secara gratis oleh siapapun.

Dengan *plugin* ini, Reasonal menawarkan efisiensi pengguna melalui *entry*, di mana informasi yang relevan tentang konten dapat diakses dan diperbarui. *Entry* dapat diedit oleh orang yang memiliki akses. Entry ini juga dibuat di *workspace* yang berbeda: *public*, *private*, dan *others*. *Public workspace* adalah *workspace* yang digunakan oleh semua orang, jadi pengguna tidak memerlukan akun untuk mengakses. Lalu, *private workspace* adalah *workspace* pribadi pengguna dan tidak ada orang lain yang dapat melihat entry. Sedangkan, *other workspace* adalah ruang

kerja untuk organisasi berbeda, hanya dapat diakses oleh orang dalam organisasi.

Meskipun *plugin* diklaim telah berfungsi dengan baik, Reasonal tetap berusaha untuk memperbaiki dan mengembangkan *plugin* tersebut. Untuk mengetahui perkembangan dari *plugin* ini, mereka membutuhkan statistik mengenai pemakaian, serta performa dari *plugin* Reasonal.

1.2. Tujuan

Tujuan dari kerja praktik ini adalah untuk menyelesaikan kewajiban kuliah kerja praktik di Institut Teknologi Sepuluh Nopember dengan bobot dua SKS. Selain itu, tujuan lainnya adalah untuk mengecek dan mengevaluasi perkembangan dari *plugin* yang dibuat oleh Reasonal.

1.3. Manfaat

Dengan grafik yang akan dibuat, diharapkan dapat menjadi dasar dari perbaikan atau pengembangan *plugin* yang akan dilakukan oleh perusahaan Reasonal DE GmbH.

1.4. Rumusan Masalah

Berikut ini rumusan masalah pada kerja praktik evaluasi kinerja plugin Reasonal:

1. Berapa *unique users* yang terdapat pada setiap *referrer*?
2. Berapa jumlah *entries* yang dibuat dan dilihat oleh pengguna di setiap *workspace*?

3. Apa saja *website* yang sering mengarahkan pengguna ke halaman *entry* Reasonal?

4. Berapa jumlah edit yang dilakukan oleh pengguna di setiap *entry*?

5. Berapa *collaborators* yang ada di setiap entry per setiap *workspace*?

1.5. Lokasi dan Waktu Kerja Praktik

Kerja praktik ini dilaksanakan pada waktu dan tempat sebagai berikut:

Lokasi	: Online
Waktu	: 12 Oktober – 3 Desember 2020
Hari Kerja	: Senin – Jumat
Jam Kerja	: Fleksibel (minimal 12 jam per minggu)

1.6. Metodologi Kerja Praktik

1.6.1. Perumusan Masalah

Dalam tahap ini kami perlu mengetahui permasalahan apa saja yang terjadi dan dapat diselesaikan atau dioptimasi. Lalu kami juga perlu mengetahui semua kebutuhan dalam permasalahan tersebut.

1.6.2. Studi Literatur

Setelah ditentukan rumusan masalah mengenai sistem yang akan dibuat, dilakukan studi literatur mengenai implementasinya. Pada tahap ini dilakukan proses pencarian, pembelajaran, dan pengumpulan informasi yang berkaitan dengan implementasi sistem yang akan dibuat. Informasi dapat diperoleh dari internet ataupun dari proyek sebelumnya yang serupa dan memungkinkan untuk diimplementasikan.

1.6.3. Analisis dan Perancangan

Tahap ini meliputi penjelasan mengenai hasil dari studi literatur yang dilakukan. Dari beberapa metode yang ditemukan saat literasi dianalisa metode mana yang paling tepat dan efektif untuk digunakan untuk menyelesaikan permasalahan. Ditentukan bahasa pemrograman yang akan digunakan, serta batasan data yang akan digunakan, sehingga dapat memunculkan hasil yang diharapkan.

1.6.4. Implementasi Sistem

Pada tahap ini dijelaskan implementasi program yang digunakan pada proses pembuatan sistem perekaman dokumen yang akan dibuat. Bagian ini meliputi penjelasan dari evaluasi kinerja *plugin*

Reasonal, dengan menggunakan bahasa pemrograman python pada google colab.

1.6.5. Pengujian dan Evaluasi

Pengujian sistem yang dilakukan merupakan pengujian terhadap grafik yang telah saya buat.

1.6.6. Kesimpulan dan Saran

Pada bab ini, dipaparkan kesimpulan yang dapat diambil dan juga saran dalam pengerjaan kerja praktik.

1.7. Sistematika Laporan

Laporan kerja praktik ini terdiri dari tujuh bab dengan rincian sebagai berikut:

1.7.1. Bab I Pendahuluan

Pada bab ini dijelaskan tentang latar belakang permasalahan, tujuan, waktu pelaksanaan, serta sistematika pengerjaan kerja praktik dan juga penulisan laporan kerja praktik.

1.7.2. Bab II Profil Perusahaan

Bab ini akan menjelaskan secara rinci tentang profil Reasonal DE GmbH, tempat saya melaksanakan kerja praktik.

1.7.3. Bab III Tinjauan Pustaka

Pada bab ini, dijelaskan mengenai tinjauan Pustaka dan literatur yang digunakan dalam penyelesaian kerja praktik di Reasonal DE GmbH.

1.7.4. Bab IV Implementasi Sistem

Pada bab ini, berisi penjelasan tahap-tahap yang dilakukan untuk proses evaluasi kinerja *plugin* Reasonal.

1.7.5. Bab V Pengujian dan Evaluasi

Pada bab ini, dijelaskan tentang hasil pengujian dan evaluasi dari grafik dan analisa yang telah dibuat selama pelaksanaan kerja praktik.

1.7.6. Bab VI Kesimpulan dan Saran

Pada bab ini, akan dipaparkan kesimpulan yang dapat diambil dan juga saran selama pengerjaan kerja praktik.

[Halaman ini sengaja dikosongkan]

BAB II

PROFIL PERUSAHAAN

2.1. Profil Reasonal DE GmbH

Reasonal DE GmbH atau yang biasa disebut Reasonal adalah sebuah perusahaan *startup* yang berlokasi di Rosenthaler str. 13, Berlin, Berlin 10119, Jerman. Reasonal diawali dengan topik penelitian dari kedua pendirinya-Behzad Tabibian dan Utkarsh Upadhyay-yang dulu sedang menempuh jenjang studi doktor di dua institusi Max-Planck di Jerman. Perusahaan ini didirkan pada tahun **2017**, dengan bantuan dari *supervisor* mereka, yaitu Benhard dan Manuel.

Dengan berbekal pengalaman Behzad dan Utkarsh di beberapa perusahaan *startup* yang besar seperti Facebook, Yahoo, dan McKinsey, mereka memutuskan untuk mendirikan perusahaan *startup* sendiri yang bernama Reasonal. Tim dari Reasonal terdiri sembilan personil, yaitu Manuel Gomez Rodriguez, PhD dan Prof. Dr. Benhard Scholkopf sebagai *co-founder*, lalu Marcus Ruebsam dan Alex Diehl selaku penasihat senior, Behzad Tabibian, PhD sebagai *chief executive officer*, kemudian Utkarsh Upadhyay, PhD selaku *chief technology officer*, Dr. Natalie Tillack sebagai *chief commercial officer*, dan Raffi Enficiaud, PhD serta Ghaith Bilbeisi selaku insyinur *machine learning*. Meskipun tidak berjumlah banyak, personil tim ini memiliki berbagai latar belakang, mulai dari teknologi hingga bisnis, sains hingga pemasaran, dan *startup* hingga perusahaan.

Perusahaan ini menciptakan sebuah *plugin* yang dapat dipasang di *browser* penggunanya. *Plugin* ini dapat menyimpan *entry* yang dibuat oleh seorang pengguna pada sebuah artikel. *Entry* ini nantinya dapat dilihat oleh pengguna lainnya, dan bahkan menambahkan komentar pada *entry* tersebut. Sehingga, kolaborasi antar pengguna dapat berjalan dengan lebih efektif dan cepat dengan adanya *plugin* yang dibuat oleh Reasonal. Cara penggunaanya pun sederhana. Pertama, pengguna dapat mengunduh *plugin* ini di bagian *extensions* pada Chrome web store. Lalu, *plugin* dapat diakses hanya dengan menekan tombol klik kanan di *mouse*, dan akan muncul opsi "*Open with Reasonal*". Opsi ini akan mengarahkan pengguna ke halaman Reasonal, dimana mereka dapat menambahkan *entry* atau komentar sesuai dengan keinginan mereka masing-masing. Reasonal berkomitmen untuk membangun *timeline smart activity* untuk setiap konten digital yang pengguna kerjakan dan mengirimkannya ke seluruh *platform* dan tim.

2.2. Logo Perusahaan

Gambar 2.2. Logo Reasonal DE GmbH

2.3. Visi Misi Perusahaan

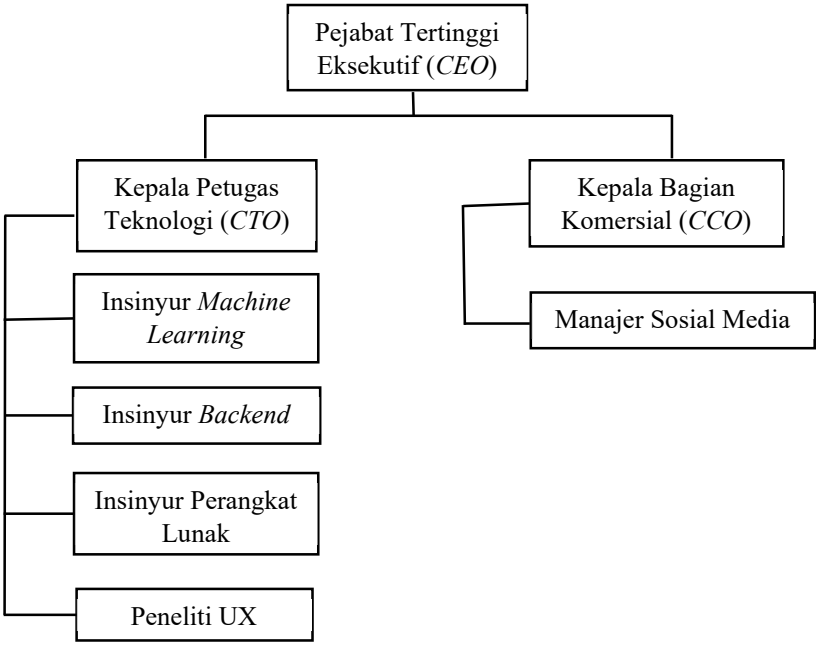
2.3.1. Visi

Menjadi perusahaan *startup* penyedia *plugin* yang unggul, terpercaya dan mendunia, demi mewujudkan manajemen pengetahuan generasi berikutnya untuk tim.

2.3.2. Misi

Memberi layanan prima dan solusi untuk pekerjaan secara remote, agar dapat menjadi produktif bagi para pihak yang terlibat.

2.4. Struktur Organisasi



Gambar 2.4. Struktur Organisasi Reasonal DE GmbH

[Halaman ini sengaja dikosongkan]

BAB III
TINJAUAN PUSTAKA

Pada bab ini, akan dijelaskan mengenai dasar teori yang digunakan selama proses kerja praktik.

3.1. Python

Python adalah bahasa pemrograman interpretatif multiguna. Bahasa pemrograman ini dibuat oleh Guido van Rossum dan dikenalkan sejak tahun 1991. Sebelum memulai untuk belajar Python dasar, akan lebih baik untuk memahami dulu apa itu Python dan bagaimana cara kerjanya.

Python merupakan bahasa pemrograman yang mudah untuk dipelajari. Tidak seperti bahasa lain yang susah untuk dibaca dan dipahami, python lebih menekankan pada keterbacaan kode agar lebih mudah untuk memahami sintaks. Hal ini membuat Python sangat mudah dipelajari baik untuk pemula maupun untuk yang sudah menguasai bahasa pemrograman lain.

Sampai saat ini bahasa pemrograman Python hampir dipakai di segala bidang seperti program CLI, Program GUI (desktop), Aplikasi Mobile, Web, IoT, Game, Program untuk Hacking, dan sebagainya. Jadi secara umum, bahasa pemrograman ini dipakai dalam pengembangan website, pengembangan software, matematika, dan system scripting.

3.2. Numpy

NumPy (Numerical Python) adalah library Python yang fokus pada scientific computing. NumPy memiliki kemampuan untuk membentuk objek N-dimensional array, yang mirip dengan list pada Python. Keunggulan NumPy array dibandingkan dengan list pada Python adalah konsumsi memory yang lebih kecil serta runtime yang lebih cepat. NumPy juga memudahkan kita pada Aljabar Linear, terutama operasi pada Vector (1-d array) dan Matrix (2-d array).

List pada Python tidak mendukung penuh kemudahan scientific computing, sebagai contoh kita akan lakukan operasi penjumlahan pada 2 list.

```
In [1]:
# list pada python
a = [1, 2, 3]
b = [4, 5, 6]
a + b

Out[1]: [1, 2, 3, 4, 5, 6]

In [2]:
# penjumlahan dilakukan iterasi
result = []
for first, second in zip(a, b):
    result.append(first + second)
result

Out[2]: [5, 7, 9]
```

Gambar 3.2. Operasi Penjumlahan 2 List

Ketika kita ingin menjumlahkan tiap elemen pada list a dan list b, hasilnya dengan operator + adalah penggabungan (concat) keduanya. Tentu tidak sesuai yang diharapkan, maka kita harus menggunakan perulangan for untuk menambahkan tiap elemen pada list a dan list b. Proses penjumlahan list yang

menggunakan perulangan for membutuhkan waktu yang lama dan tidak efisien dari sisi penulisan code.

3.3. Pandas

Pandas adalah sebuah librari berlisensi BSD dan open source yang menyediakan struktur data dan analisis data yang mudah digunakan dan berkinerja tinggi untuk bahasa pemrograman Python.

Dengan kata lain, Pandas adalah librari analisis data yang memiliki struktur data yang diperlukan untuk membersihkan data mentah ke dalam sebuah bentuk yang cocok untuk analisis (yaitu tabel). Pandas melakukan tugas penting seperti menyelaraskan data untuk perbandingan dan penggabungan set data, penanganan data yang hilang, dll, itu telah menjadi sebuah librari de facto untuk pemrosesan data tingkat tinggi dalam Python (yaitu statistik). Pandas pada mulanya didesain untuk menangani data finansial, dikarenakan altenatif umum adalah menggunakan spreadsheet (misalnya Microsoft Excel).

Struktur data dasar pandas dinamakan DataFrame, yaitu sebuah koleksi kolom berurutan dengan nama dan jenis, dengan demikian merupakan sebuah tabel yang tampak seperti database dimana sebuah baris tunggal mewakili sebuah contoh tunggal dan kolom mewakili atribut tertentu. Harus dicatat di

sini bahwa elemen dalam berbagai kolom mungkin berapa jenis yang berbeda.

Dengan adanya fitur dataframe memudahkan untuk membaca sebuah file dan menjadikannya table, kita juga dapat mengolah suatu data dengan menggunakan operasi seperti join, distinct, group by, agregasi, dan teknik lainnya yang terdapat pada SQL. Banyak format file yang dapat dibaca menggunakan Pandas, seperti file .txt, .csv, .tsv dan lainnya.

3.4. DataFrame

DataFrame merupakan tabel data yang terdapat kolom dan baris, dimana nilai-nilai yang terdapat di dalamnya dapat berupa tipe berbeda seperti numeric, string, boolean, dll. DataFrame mirip dengan data 2-dimensi dengan adanya baris dan kolom. Selain itu, DataFrame bisa dikatakan gabungan dari dictionary objek Series yang memiliki indeks yang sama.

Terdapat berbagai macam cara untuk membentuk objek DataFrame. Salah satu cara yang biasa dilakukan untuk membentuk objek DataFrame dengan menggunakan data masukan berupa dictionary.

```
In [15]:
# membuat dataframe
data = {'kota': ['semarang', 'semarang', 'semarang',
                 'bandung', 'bandung', 'bandung'],
        'tahun': [2016, 2017, 2018, 2016, 2017, 2018],
        'populasi': [1.5, 2.1, 3.2, 2.3, 3.2, 4.5]}
frame = pd.DataFrame(data)
frame
```

Out[15]:

	kota	tahun	populasi
0	semarang	2016	1.5
1	semarang	2017	2.1
2	semarang	2018	3.2
3	bandung	2016	2.3
4	bandung	2017	3.2
5	bandung	2018	4.5

```
In [16]:
# cek tipe
type(frame)
```

```
Out[16]: pandas.core.frame.DataFrame
```

Gambar 3.4.1. Pembentukan Objek DataFrame

Kita dapat menggunakan fungsi shape untuk mengetahui jumlah baris dan kolom dari DataFrame. Fungsi shape pada DataFrame memiliki hasil keluaran yang sama dengan fungsi shape pada NumPy, dimana menghasilkan keluaran sebuah tuple dari jumlah baris dan jumlah kolom.

```
In [17]:
# cek shape dari frame
frame.shape
```

```
Out[17]: (6, 3)
```

```
In [18]:
# cek info dari frame
frame.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6 entries, 0 to 5
Data columns (total 3 columns):
kota        6 non-null object
tahun       6 non-null int64
populasi    6 non-null float64
dtypes: float64(1), int64(1), object(1)
memory usage: 224.0+ bytes
```

Gambar 3.4.2. Penggunaan Fungsi Shape

Fungsi info() sangat berguna untuk mengetahui keterangan dari objek DataFrame yang kita buat seperti index dari DataFrame lengkap dengan range dari index, jumlah kolom beserta informasi tiap kolom untuk null data dan tipe data, dan jumlah total penggunaan memory pada tiap kolom dalam satuan bytes. Saat kita input data dictionary pada DataFrame, kita tidak perlu mendefinisikan tipe data untuk masing-masing kolom, karena secara otomatis pandas akan memberikan tipe data sesuai dengan values untuk tiap kolom, meskipun kita juga bisa mendefinisikan tipe data secara manual.

Misalkan kita memiliki objek DataFrame yang memiliki baris hingga jutaan, kita tidak ingin menampilkan data secara keseluruhan karena akan menghabiskan memory. Kita dapat menggunakan fungsi head() dan tail() untuk menampilkan data secara default untuk 5 data teratas dan 5 data terbawah.

In [19]:
```
# menampilkan data 5 teratas
frame.head()
```

Out[19]:

	kota	tahun	populasi
0	semarang	2016	1.5
1	semarang	2017	2.1
2	semarang	2018	3.2
3	bandung	2016	2.3
4	bandung	2017	3.2

In [20]:
```
# menampilkan data 5 terbawah
frame.tail()
```

Out[20]:

	kota	tahun	populasi
1	semarang	2017	2.1
2	semarang	2018	3.2
3	bandung	2016	2.3
4	bandung	2017	3.2
5	bandung	2018	4.5

Gambar 3.4.3. Fungsi head() dan tail() untuk Menampilkan Data

Seperti pada Series, kita juga dapat mengakses kolom pada DataFrame menggunakan fungsi columns dan untuk mengakses indeks dari DataFrame menggunakan fungsi index. Jika ingin mendapatkan data pada DataFrame secara keseluruhan menggunakan fungsi values yang akan menghasilkan output berupa array 2-dimensi sesuai dengan jumlah baris dan kolom.

```
In [22]:
# column pada dataframe
frame.columns

Out[22]: Index(['kota', 'tahun', 'populasi'], dtype='object')

In [23]:
# index pada dataframe
frame.index

Out[23]: RangeIndex(start=0, stop=6, step=1)

In [29]:
# mendapatkan keseluruhan data
frame.values

Out[29]: array([['semarang', 2016, 1.5],
                ['semarang', 2017, 2.1],
                ['semarang', 2018, 3.2],
                ['bandung', 2016, 2.3],
                ['bandung', 2017, 3.2],
                ['bandung', 2018, 4.5]], dtype=object)
```

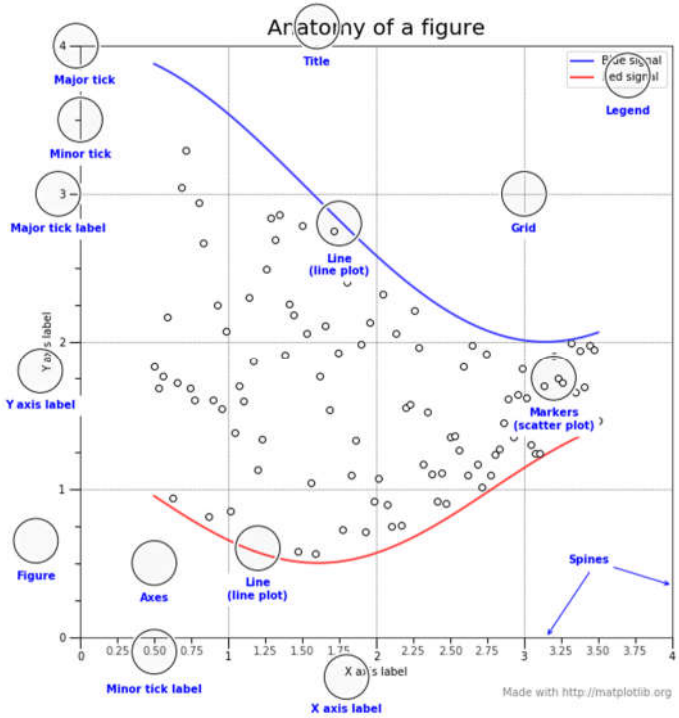
Gambar 3.4.4. Fungsi Values untuk Menghasilkan Array 2D

3.5. Matplotlib

Matplotlib adalah library Python yang fokus pada visualisasi data seperti membuat plot grafik. Matplotlib pertama kali diciptakan oleh John D. Hunter dan sekarang telah dikelola oleh tim developer yang besar. Awalnya matplotlib dirancang untuk menghasilkan plot grafik yang sesuai pada publikasi jurnal atau artikel ilmiah. Matplotlib dapat digunakan dalam skrip Python, Python dan IPython shell, server aplikasi web, dan beberapa toolkit graphical user interface (GUI) lainnya.

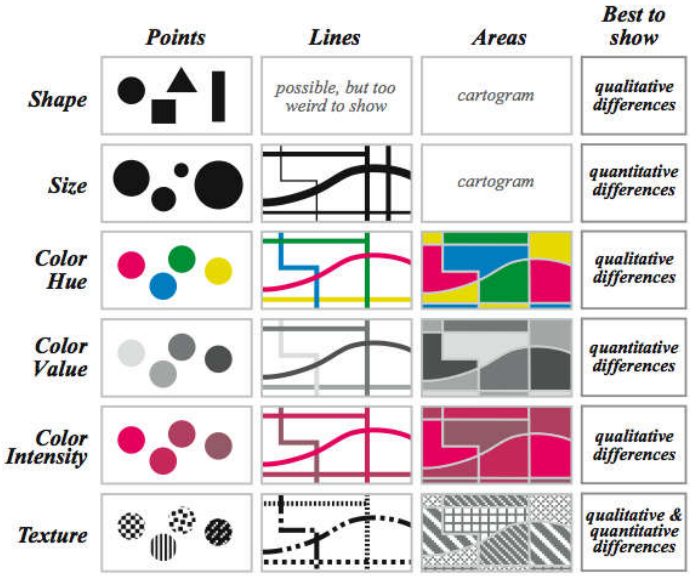
Visualisasi dari matplotlib adalah sebuah gambar grafik yang terdapat satu sumbu atau lebih. Setiap sumbu memiliki sumbu horizontal (x) dan sumbu vertikal (y), dan data yang direpresentasikan menjadi warna dan glyphs seperti marker (contohnya bentuk lingkaran) atau lines (garis) atau poligon.

Gambar di bawah menunjukkan bagian-bagian dari visualisasi matplotlib dibuat oleh Nicolas P. Rougier.



Gambar 3.5.1. Visualisasi Matplotlib oleh Nicolas P. Rougier

Hal yang penting dalam visualisasi data adalah penentuan warna, tekstur, dan style yang menarik untuk dilihat dan representatif terhadap data. Seorang Cartographer yaitu Jacques Bertin mengembangkan rekomendasi berikut untuk pemilihan informasi visual yang cocok, dan kita dapat menerapkannya menggunakan matplotlib.



Gambar 3.5.2. Pilihan Informasi Visual dalam Matplotlib

Untuk memulai menggunakan matplotlib, lakukan import terlebih dahulu library matplotlib.pyplot as plt. Penggunaan as disini, artinya kita menggantikan pemanggilan fungsi pyplot pada matplotlib dengan prefix plt untuk proses berikutnya. Disini terdapat magic command %matplotlib inline, untuk pengaturan pada backend matplotlib agar setiap grafik ditampilkan secara 'inline', yaitu akan ditampilkan langsung pada cell notebook.

```
In [1]:
1 # import library
2 import matplotlib.pyplot as plt
3 %matplotlib inline
```

Gambar 3.5.3. Mengimport matplotlib ke Notebook

3.6. Seaborn

Untuk instalasi Python standar, kita bisa menginputkan perintah pip install seaborn pada terminal GNU/Linux dan MacOs atau PowerShell Windows. Perlu diingat bahwa Seaborn sangat bergantung pada hal-hal berikut :

- Python 2.7+ atau Python 3.4+
- NumPy
- SciPy
- pandas
- Matplotlib

Jadi, sebaiknya library tersebut sudah terlebih dahulu terinstall didirektori kerja Python sebelum menggunakan Seaborn.

Hal petama yang dilakukan yaitu dengan mengimport library NumPy dan pandas, yang mana library ini adalah yang sangat populer untuk mengelola dataset bersifat relasional (format tabel). Perintahnya adalah sebagai berikut :

```
import pandas as pd
import numpy as np
```

Kemudian lakukan import induk Seaborn, siapa ? tentu saja Matplotlib :-) agar kita dapat akses untuk menggunakan Seaborn.

```
import matplotlib.pyplot as plt
```

Selanjutnya adalah Seaborn itu sendiri.

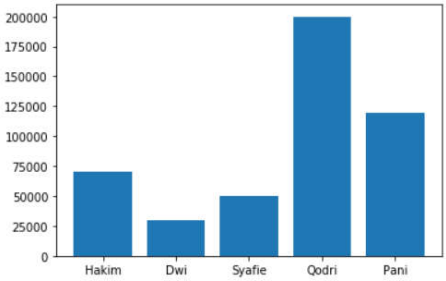
```
import seaborn as sns
```

Agar lebih mudah untuk pembelajaran, kita mulai dari jenis plot tunggal. Ada beberapa jenis grafik plot variabel tunggal yang umum dan juga sangat penting untuk kita ketahui yaitu, line plots, bar plots, histogram/density plots dan box whisker plots.

Untuk membuat bar plot sederhana, dapat menggunakan metode sebagai berikut.

```
nama = ["Hakim", "Dwi", "Syafie", "Qodri","Pani"]
isi_dompet = [70000, 30000, 50000, 200000, 120000]

# Membuat barplot menggunakan matplotlib
plt.bar(nama, isi_dompet)
plt.show()
```

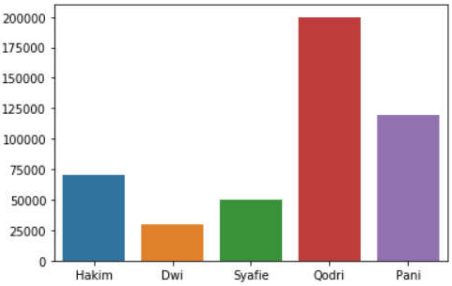
Gambar 3.6.1. Pembuatan Barplot dengan Menggunakan Matplotlib



Gambar 3.6.2. Hasil Barplot Matplotlib

```
# Membuat barplot menggunakan seaborn
sns.barplot(x=nama, y=isi_dompet)
plt.show()
```

Gambar 3.6.3. Penggunaan Barplot Seaborn



Gambar 3.6.4. Hasil Barplot Menggunakan Seaborn

Setelah semua tahap dilakukan, maka akan menghasilkan bar plot dengan data seperti di atas.

3.7. Google Colab

Google Colab (atau cukup disebut Colab) adalah produk dari Google Research. Colab adalah executable document, yang bisa kamu gunakan untuk menulis, menyimpan, serta membagikan program yang telah kamu tulis melalui Google Drive. Jika kamu familiar dengan Jupyter Notebook, maka Colab bisa dikatakan sebagai Notebook yang disimpan pada Google Drive.

Colab merupakan cloud-based runtime, yang dijalankan menggunakan browser (Chrome, Firefox dan Safari). Colab memungkinkan kalian menjalankan kode Python tanpa memerlukan proses instalasi dan setting lainnya pada komputer pribadi kita. Semua akan diserahkan ke cloud. Kita dapat juga menggunakan berbagai fungsionalitas yang dimiliki oleh Python serta memanfaatkan built-in library yang disediakan oleh Colab. Sebagai contoh, jika kita ingin membuat visualisasi data, kita dapat menggunakan contoh koding program yang telah disediakan oleh Colab dan memasukkan cuplikan koding tersebut ke project kita. Terlebih dari itu semua, layanan Colab diberikan secara gratis. Dengan hanya memiliki akun Google, kita dapat menggunakan Colab.

Kode pada Colab dieksekusi pada mesin virtual tersendiri, bersesuaian dengan akun Google yang kita miliki. Karena fasilitas ini sifatnya gratis, maka tentu Google malakukan pengaturan ataupun pembatasan dalam

penggunaan Colab. Mesin virtual ini dapat terhenti jika kita dalam posisi idle dalam rentang waktu tertentu, dan maksimum menjalankan Notebook saat ini dilaporkan selama maksimum 12 jam.

[Halaman ini sengaja dikosongkan]

BAB IV

IMPLEMENTASI SISTEM

Pada bab ini akan menjelaskan tahap implementasi yang dilakukan pada data yang diberikan oleh perusahaan Reasonal DE GmbH selama kerja praktik.

4.1. Menyambungkan ke Google Drive

Pertama, saya menyambungkan *notebook* yang saya buat di google colab ke google drive saya, agar dapat memuat data yang dibutuhkan.

```
START
Connect Drive to Google Colab
GET content
```

Pseudocode 4.1. Menyambungkan ke Google Drive

4.2. Import Library

Langkah selanjutnya adalah meng*import* library yang dibutuhkan, agar dapat menghasilkan output yang diharapkan ketika program dijalankan.

```
import all libraries
    import parse
    import gzip
    import json
    import multiprocessing
    import numpy
    import pandas
    import matplotlib.pyplot
    import matplotlib
    import urllib.parse
    import utils
    import math
    import seaborn
```

```
    import HTML
    import date, timedelta
    import Path
```

Pseudcode 4.2. Mengimport Semua Library yang Dibutuhkan

4.3. Memuat Data dari Google Drive

Setelah *notebook* tersambungkan ke google drive, maka saya dapat memuat data yang saya butuhkan, yaitu file "log_dump1.gz".

```
Find path /content/drive/My Drive/Career Semester Internship/
GET file log_dump1.gz
log<-log_dump1.gz
PRINT log length
```

Pseudocode 4.3. Memuat Data dari Google Drive

4.4. Memunculkan Dataframe

Untuk memunculkan *dataframe* dalam bentuk tabel, kita harus mendefinisikan fungsi yang berisi elemen-elemen yang ada pada *dataframe*. Lalu, *dataframe* dimuat dengan *library* pandas, dan akan mengeluarkan output sebagai berikut:

Tabel 4.4. Tabel Dataframe

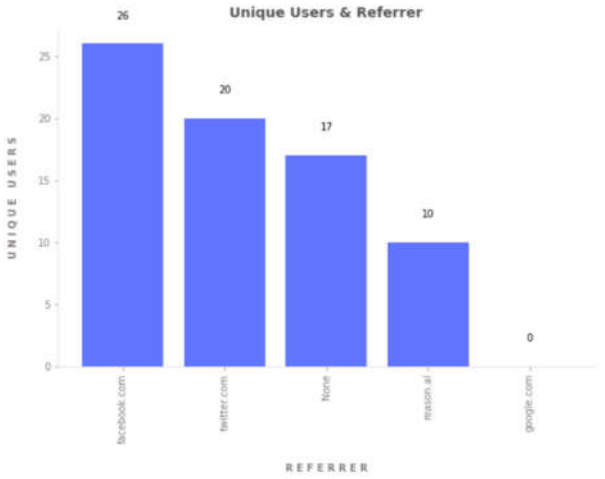
```
run function "nginx_log_worker"
SET more_data
    set path to none
    set id to none
    set org_id to 1
    set link to none
    set source to none
    set title to none
    set lang to none
    set utm_source to none

IF api_args:
            url_parse <- UP.urlparse(api_args['url'])

            more_data['path'] <- url_parse.path

            qs <- UP.parse_qs(url_parse.query)

            IF '_id' in qs:
                 set more_data['_id'] to qs['_id']
            ENDIF

            IF 'org_id' in qs:
                 set more_data['org_id'] to qs['org_id']
            ENDIF

            IF 'link' in qs:
                 set more_data['link'] to qs['link']
            ENDIF

            IF '_lang' in qs:
```

```
                set more_data['lang'] to qs['_lang']
            ENDIF

            IF 'metadata' in qs:
                set metadata to qs['metadata']
                set more_data['metadata_source'] to metadata['source']
                set more_data['metadata_title'] to metadata['title']
            ENDIF

            IF 'utm_source' in qs:
               set more_data['utm_source'] to qs['utm_source']
            ENDIF
        ENDIF

        ELSE
            output "Error while processing: "
            return None

set aug_nginx_logs to pool.map(_nginx_log_worker, nginx_logs)

FOR x in aug_nginx_logs
    if x is not None
        data = x

df<-DataFrame(data)
SHOW table df
```

Pseudocode 4.4. Memunculkan Tabel Dataframe

4.5. Unique Users & Referrer

Pada grafik pertama ini, akan menunjukkan tentang berapa jumlah *user* yang ada di setiap *referrer*-nya. *Referrer* yang dimaksud adalah alamat *website* yang mengarahkan *user* ke *website* Reasonal.



Gambar 4.5. Grafik Unique Users di setiap referrer

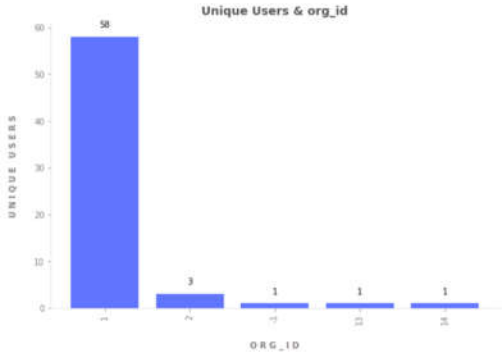
```
set lang_date to df[referer_domain]
group by referer_domain

IF ascending = False, drop = True
    lang_date<-sort values of '_id'

FOR index, row in lang_date
    PRINT black marker on '_id'
    center marker

PRINT title "Unique Users & Referrer"
PRINT x label "referrer"
PRINT y label "unique users"

SHOW bar graph
```

Pseudocode 4.5. Membuat Grafik Unique Users di setiap referrer

4.6. Unique Users & org_id

Grafik kedua ini akan menunjukkan tentang berapa jumlah *user* yang ada di setiap org_id-nya. Org_id adalah

data organisasi atau tim yang mengedit *entry* di plugin reasonal.



Gambar 4.6. Grafik Unique Users per org_id

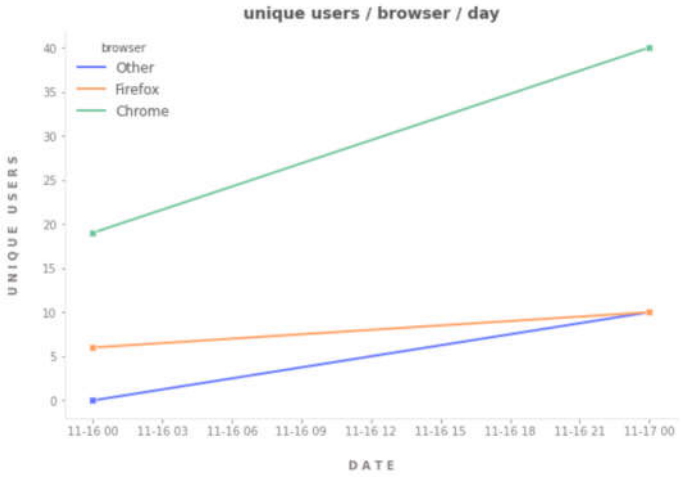
```
set df_public to df[org_id][1]
set df_reasonal to df[org_id][2]
set df_private to df[org_id][None]

user_org<-group by dataframe 'org_id'
    FOR ascending=False, drop=True
        sort values '_id'

    FOR index, row in user_org
        PRINT black marker on '_id'
        center marker

PRINT title "Unique Users & org_id"
PRINT x label "org_id"
PRINT y label "unique users"

SHOW bar graph
```

Pseudocode 4.6. Membuat Grafik Unique Users per org_id

4.7. Unique Users & Browsers

Berbeda dengan grafik sebelumnya, grafik ini akan menggunakan *lineplot*. Grafik akan menunjukkan jumlah

unique users per *browser* yang menyediakan layanan. *Browser* yang tersedia adalah google chrome, firefox, dan *other* atau lainnya.



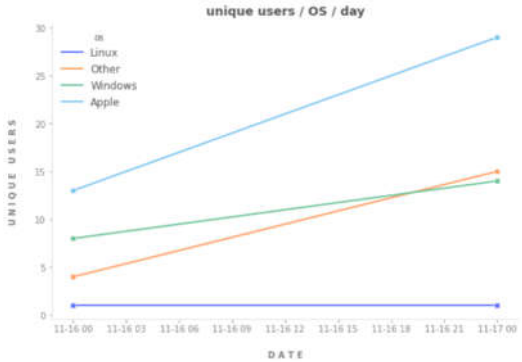
Gambar 4.7. Grafik Unique Users per Browsers

```
df['browser']<-df http_user_agent
IF x.find('Chrome') is not equal to -1
    set Chrome
    ELSE
    IF x.find('Firefox') is not equal to -1
        set Firefox
        ELSE
        set Other

set lang_date to df
    group by 'date', 'browser'
    sort values '_id'

PRINT x label "date"
PRINT y label "unique users"
PRINT title "unique users / browsers / day"

SHOW line graph
```

Pseudocode 4.7. Membuat Grafik Unique Users per Browsers

4.8. Unique Users & OS

Grafik pada poin 4.8 akan menyerupai grafik di poin 4.7, akan tetapi berbeda dari data yang dipakai. Data yang dipakai pada grafik ini adalah *unique users* di setiap OS (*operating system*) yang menyediakan layanan Reasonal, mulai dari linux, windows, apple, hingga others.



Gambar 4.8. Grafik Unique Users per OS

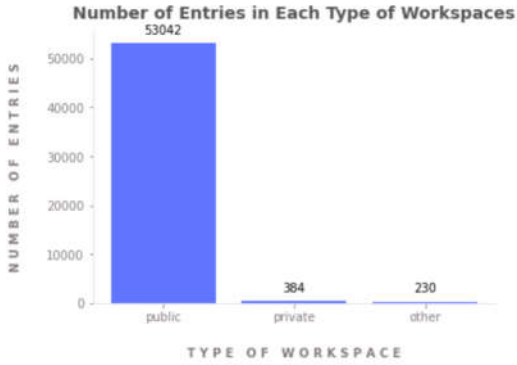
```
df['os']<-df http_user_agent
IF x.find('Windows') is not equal to -1
    set Windows

    ELSE
    IF x.find('Apple') is not equal to -1
        set Apple

        ELSE
        IF x.find('Linux') is not equal to -1
            set Linux

            ELSE
            set Other

set lang_date to df
    group by 'date', 'os'
    sort values '_id'
```

```
PRINT x label "date"
PRINT y label "unique users"
PRINT title "unique users / os / day"

SHOW line graph
```

Pseudocode 4.8. Membuat Grafik Unique Users per OS

4.9. Jumlah Entri di Setiap Workspace

Workspace dari fitur Reasonal terbagi menjadi 3, yaitu *public*, *private*, dan *other*. Public merupakan *workspace* yang dipakai oleh user secara publik, lalu private adalah *workspace* yang hanya diakses oleh user tersebut, dan other adalah *workspace* yang dapat diakses oleh tim pengembang aplikasi. Dengan itu, maka *workspace* public akan memiliki jumlah entri paling banyak, dengan total 53042 entri.



Gambar 4.9. Grafik Jumlah Entri di Setiap Workspace

```
bar graph<-df[type_of_space], [numentries]

    FOR index, row in dataframe
        PRINT black marker on 'numentries'
        center marker
```

```
PRINT x label "type of workspace"
PRINT y label "number of entries"
PRINT title "Number of Entries in Each Type of Workspaces"

SHOW bar graph
```

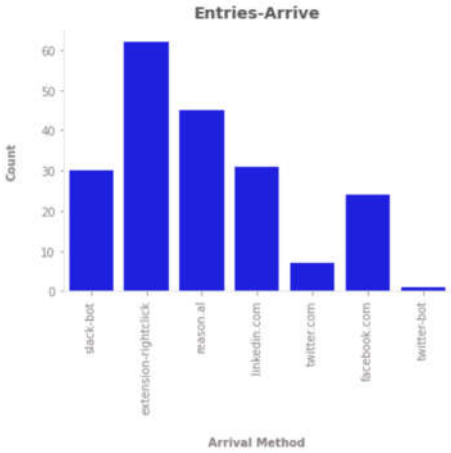
Pseudocode 4.9. Membuat Grafik Jumlah Entri per Workspace

4.10. Entri dari Setiap Website

Akses ke reasonal memiliki berbagai jenis sumber, mulai dari bot slack dan twitter, extension dengan menggunakan klik kanan mouse, reason.al, linkedin, twitter, hingga facebook. Dengan tabel dan grafik berikut ini, dapat dibuktikan bahwa *extension* dengan menggunakan klik kanan menghasilkan entri paling banyak.

```
extension-rightclick    62
reason.al               45
linkedin.com            31
slack-bot               30
facebook.com            24
twitter.com              7
twitter-bot              1
Name: utm_source, dtype: int64
```

Gambar 4.10.1. Jumlah Entri dari Setiap Website



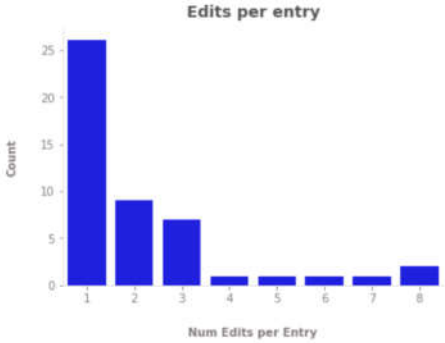
Gambar 4.10.2. Grafik Jumlah Entri per Website

```
df<-unique referer_domain
df<-value utm_source

FOR x = 'utm_source'
    set data to df

    SHOW bar graph (data)
    set color to blue
    PRINT title "Entries-Arrive"

PRINT x label "Arrival Method"
PRINT y label "Count"
```

Pseudocode 4.10. Membuat Grafik Jumlah Entri per Website

4.11. Edits per Entry

Di setiap *entry* memiliki jumlah edit yang bervariasi, mulai dari 1 hingga 8 kali edit. Pada grafik di bawah, dapat diketahui bahwa edit sebanyak 1 kali paling banyak dilakukan oleh user. Selain itu, terdapat tabel yang menunjukkan berapa kali sebuah *link* tersebut diedit.



Gambar 4.11.1. Grafik Jumlah Edit per Entri

	link	num edits
0	https://en.wikipedia.org/wiki/Amsterdam	8
1	https://theblondeabroad.com/ultimate-amsterdam...	8
2	https://archive.org/details/novelsofmargaret00...	7
3	https://www.google.com/flights	6
4	https://en.wikipedia.org/wiki/Madeira	5

Tabel 4.11. Tabel Jumlah Edit di Setiap Link

```
df_new<-df_edits_oth
rename column 'index' to 'link'
rename column 'link' to 'num edits'

set data to df_new

SHOW bar graph(data)
    set color to blue
    PRINT title "Edits per entry"

PRINT x label "Num Edits per Entry"
PRINT y label "Count"
```

Pseudocode 4.11. Membuat Grafik Jumlah Edit per Entri

[Halaman ini sengaja dikosongkan]

BAB V
PENGUJIAN DAN EVALUASI

Bab ini menjelaskan tahap uji coba dan evaluasi yang dilakukan terhadap grafik yang telah dibuat.

5.1. Tujuan Pengujian

Pengujian dilakukan terhadap grafik untuk menguji kesesuaian dan ketepatan pembuatan grafik dari data yang telah diberikan oleh Reasonal DE GmbH.

5.2. Kriteria Pengujian

Penilaian atas pencapaian tujuan pengujian grafik didapatkan dengan memperhatikan kesesuaian grafik dengan sumber data, yaitu *file* "log_dump1.gz".

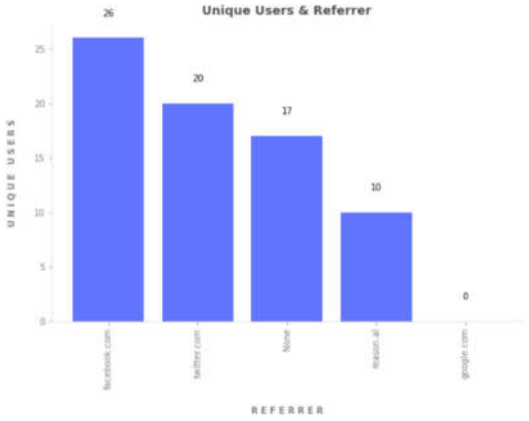
5.3. Skenario Pengujian

Skenario pengujian melibatkan seorang pembimbing lapangan yaitu Dr. Natalie Tillack beserta Utkarsh Upadhyay untuk mencocokan grafik yang telah dibuat, dengan tren serta data yang telah diperoleh oleh perusahaan Reasonal DE GmbH selama tiga minggu terakhir.

Pengujian dilakukan untuk semua grafik yang telah dibuat sebagai berikut:

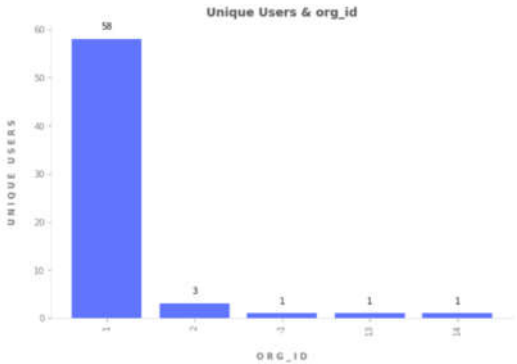
1. Jumlah *unique users* yang terdapat pada grafik dinilai sesuai dengan tren yang diketahui oleh

pembimbing lapangan. *Website* facebook memiliki jumlah pengarahan terbanyak ke *plugin* Reasonal, karena banyaknya jumlah artikel berekstensi Reasonal yang dibagikan pada website tersebut.



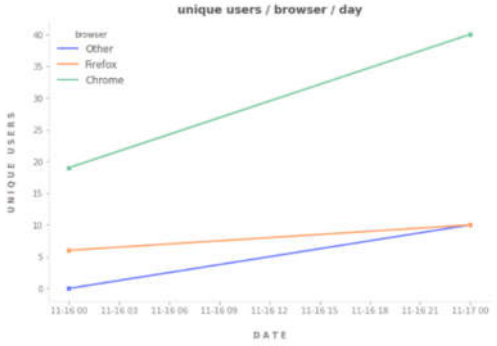
Gambar 5.3.1 Pengujian Grafik Users dan Referrer

2. Jumlah *unique users* terbanyak dimiliki oleh pengguna dengan ID "1", dimana ID "1" merupakan kode untuk publik atau pengguna umum.



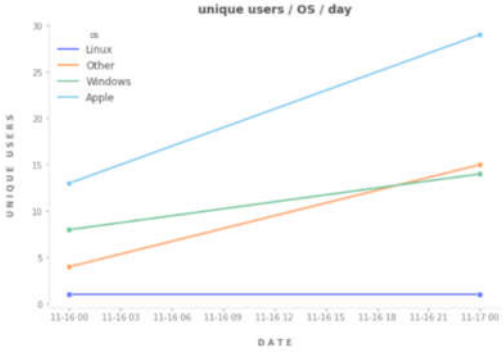
Gambar 5.3.2. Pengujian Grafik Users per org_id

3. Dalam grafik berikutnya, pengujian dilakukan dengan menggunakan data *unique users* dan *browser* yang digunakan. Grafik menunjukkan bahwa *browser* chrome lebih banyak dipakai oleh pengguna. Hasil ini *valid* dan sesuai dengan data yang dimiliki oleh pembimbing lapangan.



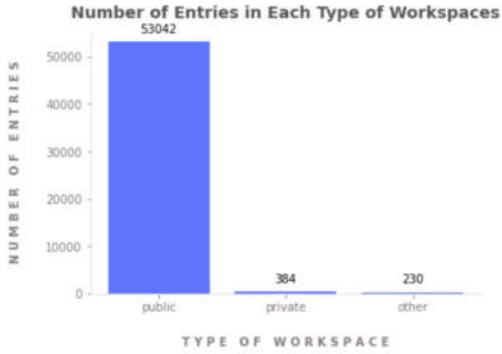
Gambar 5.3.3. Pengujuan Grafik Unique Users per Browser

4. Selain *browser*, pengujian juga dilakukan dengan data *unique users* dan OS (*operating system*). Hasil menunjukkan bahwa OS Apple mendominasi pemakaian *plugin* Reasonal.



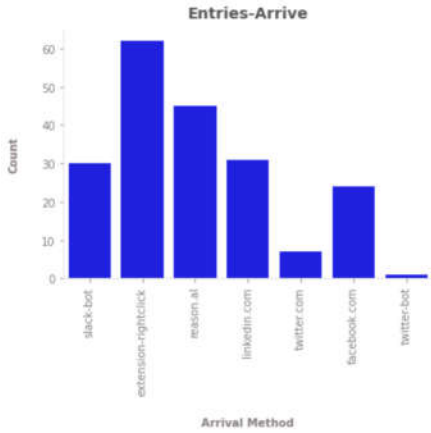
Gambar 5.3.4. Pengujian Grafik Unique Users per OS

5. Pengujian berikutnya dilakukan dengan grafik yang menunjukkan jumlah entri per *workspace*. Pembimbing lapangan menilai bahwa hasil grafik telah sesuai dengan data yang ada, karena jumlah entri yang diakses oleh publik tentunya memiliki jumlah yang lebih banyak daripada *workspace* lainnya.



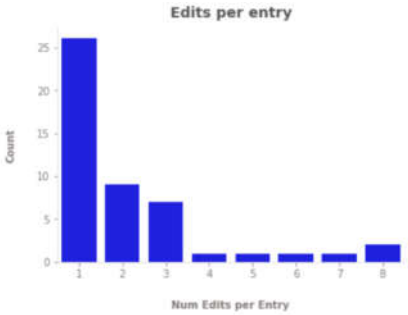
Gambar 5.3.5. Pengujian Grafik Entri per Workspace

6. Dalam grafik berikutnya, grafik entri di setiap *website* asal telah sesuai dengan data perusahaan Reasonal. Sebagai fitur yang tersedia pada menu *extension plugin* di chrome, tentunya membuat pengguna lebih banyak menggunakan klik kanan atau *right click* untuk mengakses entri pada sebuah artikel.



Gambar 5.3.6. Pengujian Grafik Entri Berdasarkan Asal

7. Jumlah edit per entri pada grafik selanjutnya telah sesuai dengan data yang dipunya oleh pembimbing lapangan. Mayoritas pengguna plugin Reasonal hanya melakukan satu kali edit di entri yang dibuat. Sehingga, jumlah satu edit memiliki jumlah tertinggi pada grafik berikut ini.



Gambar 5.3.7. Pengujian Grafik Edit per Entri

5.4. Evaluasi Pengujian

Berdasarkan pengujian yang telah dipaparkan di atas, maka pembimbing lapangan telah menyesuaikan dan menyetujui grafik yang telah dibuat. Berikut ini adalah rangkuman dari pengujian yang dilakukan:

Grafik	Deskripsi Grafik
Unique Users & Referrer	Terpenuhi
Unique Users & Org_id	Terpenuhi
Unique Users & Browser	Terpenuhi
Unique Users & OS	Terpenuhi
Entri per Workspace	Terpenuhi
Entri berdasarkan Asal Akses	Terpenuhi
Edit per Entri	Terpenuhi

Tabel 5.4. Tabel Evaluasi Pengujian Grafik

Dengan hasil pengujian pada tabel di atas, maka dapat disimpulkan bahwa secara keseluruhan grafik telah terpenuhi.

BAB VI
KESIMPULAN DAN SARAN

6.1. Kesimpulan

Kesimpulan yang didapat setelah melaksanakan pembuatan grafik dan Analisa performa *plugin* pada kegiatan kerja praktik di Reasonal DE GmbH adalah sebagai berikut:

- Dengan adanya grafik yang telah dibuat, maka mempermudah perusahaan Reasonal DE GmbH untuk mengetahui evaluasi atas performa *plugin* yang telah dibuat.
- Penggunaan bahasa pemgoraman python untuk membuat grafik yang diinginkan dinilai sangat cocok dikarenakan python merupakan bahasa pemgoraman yang mudah dipahami dan banyak dipakai oleh khalayak ramai.

6.2. Saran

Saran untuk pembuatan grafik dan analisa kinerja plugin Reasonal adalah sebagai berikut:

- Dilakukan pembaruan data dengan format ".gz" di setiap minggunya, agar data yang dipakai selalu baru dan *up-to-date*.
- Analisa performa *plugin* dapat dilakukan dengan visualisasi dalam model lain.

DAFTAR PUSTAKA

- K, Yasin. 2019. Belajar Python Pemula: Pengenalan Dasar. https://www.niagahoster.co.id/blog/belajar-python/ (diakses tanggal 5 Desember 2020)
- Muhardian, Ahmad. 2018. Belajar Pemrograman Python: Pengenalan Dasar Python dan Persiapan Awal. https://www.petanikode.com/python-linux/ (diakses tanggal 5 Desember 2020)
- Rohman, Yasir. 2019. Pengenalan NumPy, Pandas, Matplotlib. https://medium.com/@yasirabd/pengenalan-numpy-pandas-matplotlib-b90bafd36c0 (diakses tanggal 11 Desember 2020)
- Willems, Karlijn. 2019. Python Numpy Tutorial. https://www.datacamp.com/community/tutorials/python-numpy-tutorial (diakses tanggal 15 Desember 2020)
- Mccaslin, Hakim. 2019. Apa itu Seaborn? https://medium.com/@HakimMccaslin/apa-itu-seaborn-234a224d946f (diakses tanggal 15 Desember 2020)
- Hunter, John. 2020. Pyplot tutorial. https://matplotlib.org/users/pyplot_tutorial.html (diakses tanggal 18 Desember 2020)
- Waskom, Michael. 2020. An introduction to seaborn. https://seaborn.pydata.org/introduction.html (diakses tanggal 18 Desember 2020)
- Wardana, Kusuma. 2020. [Deep Learning] Apa itu Google Colab? https://tutorkeren.com/artikel/deep-learning-apa-itu-google-colab.htm (diakses tanggal 20 Desember 2020)

LAMPIRAN

LAMPIRAN 1. Menyambungkan ke Google Drive

Kode untuk memuat data yang dibutuhkan, dari google drive.

```
from google.colab import drive
drive.mount('/content/drive/')
```

```
Mounted at /content/drive/
```

```
%matplotlib inline
%load_ext autoreload


%autoreload 2
```

LAMPIRAN 2. Import Library

Kode untuk mengimport semua library yang dibutuhkan.

```
import parse
import gzip
import json
import multiprocessing as MP
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib
import urllib.parse as UP
import utils
import math
import seaborn as sns
from IPython.core.display import HTML
from datetime import date, timedelta
from pathlib import Path

plt.style.use("https://drive.google.com/uc?id=1YcJZAoChsZQlVlqRoFPJsvSEB43HRnyY")

fotmat_text = lambda x: " ".join(x.upper())
```

LAMPIRAN 3. Memuat Data dari Google Drive

Kode untuk memuat data dari google drive.

LAMPIRAN 4. Memunculkan Dataframe

Kode untuk memunculkan tabel dataframe.

```
nginx_logs = []
for ret in rets:
    nginx_logs.extend(ret[1])
```

```
%%time

import json

api_pat = parse.compile('GET {url} HTTP/{}')

def _nginx_log_worker(log):
    more_data = {
        'path': None,
        '_id': None,
        'org_id': 1,
        'link': None,
        'source': None,
        'title': None,
        'lang': None,
        'utm_source' : None,
    }
    try:
        api_args = api_pat.parse(log['request'])
        if api_args:
            url_parse = UP.urlparse(api_args['url'])

            more_data['path'] = url_parse.path

            qs = UP.parse_qs(url_parse.query)

            if '_id' in qs:
                more_data['_id'] = qs['_id'][0]

            if 'org_id' in qs:
                more_data['org_id'] = qs['org_id'][0]

            if 'link' in qs:
                more_data['link'] = UP.unquote(qs['link'][0])

            if '_lang' in qs:
                more_data['lang'] = qs['_lang'][0]

            if 'metadata' in qs:
                metadata = json.loads(qs['metadata'][0])
                more_data['metadata_source'] = UP.unquote(metadata.get('source', '###'))
                more_data['metadata_title'] = UP.unquote(metadata.get('title', '###'))

            if 'utm_source' in qs:
              more_data['utm_source'] = qs['utm_source'][0]


        more_data.update(log)
        return more_data
    except Exception as e:
        print('Error while procesing:', log, ' error:', e)
        return None

with MP.Pool() as pool:
    aug_nginx_logs = pool.map(_nginx_log_worker, nginx_logs)
```

```
CPU times: user 86 ms, sys: 51.1 ms, total: 137 ms
Wall time: 936 ms
```

```
df = pd.DataFrame(data=[x for x in aug_nginx_logs if x is not None])
```

```
df
```

LAMPIRAN 5. Unique Users & Referrer

Kode untuk membuat grafik unique users di setiap referrer.

```
lang_date = df[
    ~(df.referer_domain.str.endswith('k6.io') |
      df.referer_domain.str.endswith('elasticbeanstalk.com'))
].groupby(["referer_domain"])._id.nunique().reset_index()

df["referer_domain"] = df["referer_domain"].astype(str)

lang_date = df.groupby("referer_domain")._id.nunique().reset_index()

# lang_date = lang_date.sort_values('_name')
lang_date = lang_date.sort_values(["_id"], ascending=False).reset_index(drop=True)
plt.figure(figsize=(6 * 1.6, 6))
plt.bar(lang_date["referer_domain"], lang_date["_id"])

for index, row in lang_date.iterrows():
  plt.text(index, row["_id"] + 2, row["_id"], color='black', ha="center")


# sns.catplot(lang_date['referer_domain'], lang_date['_id'], data=lang_date['_id'], kind='bar')
# sns.lineplot(lang_date['referer_domain'], lang_date['_id'], hue=lang_date['_id'], marker='s')
# sns.catplot(lang_date['referer_domain'], lang_date['_name'], data=lang_date['_name'], kind='bar', aspect=3)
plt.xticks(rotation='vertical')
_ = plt.title('Unique Users & Referrer')
_ = plt.xlabel(fotmat_text("referrer"))
_ = plt.ylabel(fotmat_text("unique users"))
```

LAMPIRAN 6. Unique Users & org_id

Kode untuk membuat grafik unique users per org_id

```
df_public = df[df["org_id"] == "1"]
df_reasonal = df[df["org_id"] == "2"]
df_private = df[df["org_id"] == "None"]

# df_public = df[df["org_name"] == "1"]
# df_reasonal = df[df["org_name"] == "2"]
# df_private = df[df["org_name"] == "None"]

df["org_id"] = df["org_id"].astype(str)

user_org = df.groupby(["org_id"])._id.nunique().reset_index()

user_org = user_org.sort_values(["_id"], ascending=False).reset_index(drop=True)
plt.figure(figsize=(6 * 1.6, 6))
plt.bar(user_org["org_id"], user_org["_id"])
#graph = sns.barplot(user_org['org_id'], user_org['_id'])

# for x,y in zip(user_org['org_id'],user_org['_id']):
#  label = "{}".format(y)
#  plt.annotate(label, # this is the text
#               (x,y), # this is the point to label
#               textcoords="offset points", # how to position the text
#               xytext=(0,10), # distance from text to points (x,y)
#               ha='center') # horizontal alignment can be left, right or center

# for v in [user_org['org_id'], user_org['_id']]:
#    plt.text(v[0], v[1], v[1]);

for index, row in user_org.iterrows():
    plt.text(index, row["_id"] + 2, row["_id"], color='black', ha="center")

_ = plt.title('Unique Users & org_id')
plt.xticks(rotation='vertical')
_ = plt.xlabel(fotmat_text("org_id"))
_ = plt.ylabel(fotmat_text("unique users"))
```

LAMPIRAN 7. Unique Users & Browsers

Kode untuk membuat grafik unique users per browsers.

```
[150] df['browser'] = df.http_user_agent.map(
          lambda x:
              "Chrome" if x.find('Chrome') != -1
              else "Firefox" if x.find('Firefox') != -1
              else "Other"
      )#.value_counts()
```

```
[151] lang_date =df.groupby(["date", "browser"])._id.nunique().reset_index()
      lang_date = lang_date.sort_values('_id')
      plt.figure(figsize=(6 * 1.6, 6))

      sns.lineplot(lang_date['date'], lang_date['_id'], hue=lang_date['browser'], marker='s')
      plt.ylabel(fotmat_text("unique users"))
      plt.xlabel(fotmat_text("date"))
      _ = plt.title("unique users / browser / day")
      #_ = plt.xticks(days.index, days.index.strftime('%d/%m'))
```

LAMPIRAN 8. Unique Users & OS

Kode untuk membuat grafik unique users per OS.

```
df['os'] =  df.http_user_agent.map(
    lambda x:
        "Windows" if x.find('Windows') != -1
        else "Apple" if x.find('Apple') != -1
        else "Linux" if x.find('Linux') != -1
        else "Other"
)
```

```
lang_date =df.groupby(["date", "os"])._id.nunique().reset_index()
lang_date = lang_date.sort_values('_id')
plt.figure(figsize=(6 * 1.6, 6))

sns.lineplot(lang_date['date'], lang_date['_id'], hue=lang_date['os'],  marker='s')
plt.ylabel(fotmat_text("unique users"))
plt.xlabel(fotmat_text("date"))
_ = plt.title("unique users / OS / day")
#_ = plt.xticks(days.index, days.index.strftime('%d/%m'))
```

LAMPIRAN 9. Jumlah Entri di Setiap Workspace

Kode untuk membuat grafik jumlah entri di setiap workspace.

```
import numpy as np
import matplotlib.pyplot as plt

plt.bar(df["type_of_space"], df["numentries"])

for index, row in df.iterrows():
  plt.text(index, row["numentries"] + 2000, row["numentries"], color='black', ha="center")

plt.ylabel(fotmat_text("Number of Entries"))
plt.xlabel(fotmat_text("type of workspace"))
plt.title('Number of Entries in Each Type of Workspaces')

# Show graphic
plt.show()
```

LAMPIRAN 10. Entri dari Setiap Website

Kode untuk memunculkan jumlah entri per website.

```
df.referer_domain.unique()
df.utm_source.value_counts()
```

Kode untuk membuat grafik jumlah entri per website.

```
sns.countplot(x= "utm_source" , data = df, color = "blue").set_title("Entries-Arrive")
plt.xlabel("Arrival Method")
plt.ylabel("Count")
plt.xticks(rotation=90)
```

LAMPIRAN 11. Edits per Entry

Kode untuk membuat tabel jumlah edit di setiap link.

```
df_new = df_edits_oth.link.value_counts().to_frame()
df_new.reset_index(level=0, inplace=True)
df_new.head()
df_new = df_new.rename(columns={'index': 'link', 'link':'num edits'})
df_new.head()
```

Kode untuk membuat grafik jumlah edit per entri.

```
sns.countplot(x= "num edits" , data = df_new, color = "blue").set_title("Edits per entry")
plt.xlabel("Num Edits per Entry")
plt.ylabel("Count")
```

BIODATA PENULIS

Nama	: Donny Fitrado
Tempat, Tanggal Lahir	: Semarang, 20 Januari 1999
Jenis Kelamin	: Laki-Laki
Agama	: Islam
Status	: Belum Menikah
Alamat	: Jl. Manyar Tirtomoyo 8 no. 16, Surabaya
Telepon	: 082244462069
Email	: donny.public@gmail.com

PENDIDIKAN FORMAL

2017 – sekarang	: S1 Teknik Informatika ITS
2014 – 2017	: SMA Kr. Cita Hati East Surabaya
2011 – 2014	: SMP Kr. Cita Hati East Surabaya
2008 – 2011	: SD Kr. Cita Hati East Surabaya
2005 – 2008	: SD YPPI II Surabaya

KEMAMPUAN

- *Web Programming* (HTML, PHP, CSS, Javascript)
- *Programming* (C, C++, Python)
- *Database Management* (MySQL)
- *Software* Perkantoran (Microsoft Word, Excel, PowerPoint)
- Bahasa (Indonesia, Inggris, Mandarin)

AKADEMIS

Kuliah	: Departemen Informatika – Fakultas Teknologi Informasi dan Komunikasi, Institut Teknologi Sepuluh Nopember Surabaya
Angkatan	: 2017
Semester	: 7 (Tujuh)
IPK	: 3.21

[Halaman ini sengaja dikosongkan]